Accelerate Your Certification with Microsoft DP-750 Practice Questions

Last chance to prepare smart! Get your hands on free Microsoft Implementing Data Engineering Solutions Using Azure Databricks exam PDF questions. Study real DP-750 dumps with verified answers and fast-track your certification success with PrepBolt DP-750 exam pdf questions and answers.

Thank you for Downloading DP-750 exam PDF Demo

https://prepbolt.com/DP-750.html

Question 1

Question Type: MultipleChoice

Case Study: Mix Questions

Mix Questions

DP-750 Mix Questions IN THIS CASE STUDY

You have an Azure Databricks workspace that is enabled for Unity Catalog and contains a Delta table named Orders.

You load the Orders table into an Apache Spark DataFrame named df.

You need to create a DataFrame that excludes rows where the order amount is null.

Solution: You run the following expression.

df.filter(df.order_amount.isNotNull())

Does this meet the goal?

Options:

A- Yes
B- No

Answer:

A

Explanation:

CORRECT ANSWE R: A - Yes.

According to Microsoft Learn on PySpark DataFrame filtering, df.filter(df.order_amount.isNotNull()) correctly removes all rows where order_amount is null and returns a DataFrame containing only rows where order_amount has a non-null value. The isNotNull() method is a Column method that evaluates to True for non-null values and False for null values, making it the correct and PySpark-idiomatic way to filter out null rows. Unlike Python's != None comparison (which uses SQL null semantics and returns NULL for null comparisons), isNotNull() explicitly handles the SQL NULL case and returns a proper boolean. This is semantically equivalent to WHERE order_amount IS NOT NULL in SQL. The filter passes all non-null rows and excludes all null rows from the resulting DataFrame.

Question 2

Question Type: MultipleChoice

Case Study: Mix Questions

Mix Questions

DP-750 Mix Questions IN THIS CASE STUDY

You have an Azure Databricks workspace that is enabled for Unity Catalog and contains a Delta table named Orders.

You load the Orders table into an Apache Spark DataFrame named df.

You need to create a DataFrame that excludes rows where the order amount is null.

Solution: You run the following expression.

df.filter(df.order_amount != None)

Does this meet the goal?

Options:

A- Yes
B- No

Answer:

B

Explanation:

CORRECT ANSWE R: B - No.

According to Microsoft Learn and PySpark documentation, comparing a DataFrame column to Python's None using the != operator (df.order_amount != None) does not work correctly in PySpark. In PySpark, null comparisons using standard Python equality/inequality operators (==, !=) with None follow SQL null semantics --- any comparison with NULL returns NULL (not True or False). As a result, df.filter(df.order_amount != None) does not correctly filter out null rows; it may return unexpected results or fail silently. The correct PySpark approach is to use the isNotNull() method: df.filter(df.order_amount.isNotNull()), or df.dropna(subset=['order_amount']). Using Python's None with comparison operators is a common mistake when transitioning from Python to PySpark.

Question 3

Question Type: MultipleChoice

Case Study: Mix Questions

Mix Questions

DP-750 Mix Questions IN THIS CASE STUDY

You have an Azure Databricks workspace that is enabled for Unity Catalog and contains a Delta table named Orders.

You load the Orders table into an Apache Spark DataFrame named df.

You need to create a DataFrame that excludes rows where the order amount is null.

Solution: You run the following expression.

df.dropna(subset=["order_amount"])

Does this meet the goal?

Options:

A- Yes
B- No

Answer:

A

Explanation:

CORRECT ANSWE R: A - Yes.

According to Microsoft Learn on PySpark DataFrame operations, df.dropna(subset=['order_amount']) removes all rows from the DataFrame where the specified column (order_amount) contains a null value. The resulting DataFrame contains only rows where order_amount is not null, which directly meets the requirement to 'create a DataFrame that excludes rows where the order_amount is null.' The dropna() method (equivalent to DataFrame.na.drop()) is the idiomatic PySpark approach for removing rows with null values in specified columns. The subset parameter limits the null check to only the order_amount column, preserving rows where other columns may be null. This is the correct and recommended approach for null row exclusion in PySpark.

Question 4

Question Type: MultipleChoice

Case Study: Mix Questions

Mix Questions

DP-750 Mix Questions IN THIS CASE STUDY

You have an Azure Databricks workspace that is enabled for Unity Catalog and contains a Delta table named Orders

You load the Orders table into an Apache Spark DataFrame named df.

You need to create a DataFrame that excludes rows where the order amount is null.

Solution: You run the following expression.

df-fillna(0, subset=['order_amount'])

Does this meet the goal?

Options:

A- Yes
B- No

Answer:

B

Explanation:

CORRECT ANSWE R: B - No.

According to Microsoft Learn on PySpark DataFrame operations, df.fillna(0, subset=['order_amount']) replaces null values in the order_amount column with the integer 0. This does NOT exclude rows where order_amount is null --- it replaces the null with 0, meaning those rows are still included in the resulting DataFrame with 0 as the order_amount value. The requirement is to 'create a DataFrame that excludes rows where the order_amount is null' --- which means null rows must be removed (dropped), not filled. The correct operation to exclude null rows is df.dropna(subset=['order_amount']) or df.filter(df.order_amount.isNotNull()). fillna is used for data imputation (replacing nulls with a default value), which is a different operation from filtering out null rows.

Question 5

Question Type: MultipleChoice

Case Study: Mix Questions

Mix Questions

DP-750 Mix Questions IN THIS CASE STUDY

You have an Azure Databricks workspace named Workspace1 that contains a lakehouse and is enabled for Unity Catalog.

You have a connection to a Microsoft SQL Server database named DB1.

You need to expose the schemas and tables of DB1 to meet the following requirements:

* The schemas and tables can be queried in Databricks.

* The schemas and tables appear alongside other Unity Catalog objects.

* The data is NOT copied into Databricks-managed storage.

Solution: You create a Lakeflow Connect pipeline and connect it to DB1. Does this meet the goal?

Options:

A- Yes

B- No

Answer:

B

Explanation:

CORRECT ANSWE R: B - No.

According to Microsoft Learn on Lakeflow Connect, Lakeflow Connect (formerly Databricks Ingestion) is a managed ingestion service that copies data FROM external databases (like SQL Server) INTO Databricks-managed Delta tables. This means Lakeflow Connect physically copies the data into Databricks storage, directly violating the requirement that 'the data is NOT copied into Databricks-managed storage.' Additionally, Lakeflow Connect creates Delta tables in Unity Catalog rather than exposing the original external database objects alongside other Unity Catalog objects in a federated manner. For the requirement to expose DB1's existing schemas and tables without data movement, Lakehouse Federation (foreign catalog) is the correct solution, not Lakeflow Connect.

Question 6

Question Type: MultipleChoice

Case Study: Mix Questions

Mix Questions

DP-750 Mix Questions IN THIS CASE STUDY

You have an Azure Databricks workspace named Workspace1 that contains a lakehouse and is enabled for Unity Catalog.

You have a connection to a Microsoft SQL Server database named DB1.

You need to expose the schemas and tables of DB1 to meet the following requirements:

* The schemas and tables can be queried in Databricks.

* The schemas and tables appear alongside other Unity Catalog objects.

* The data is NOT copied into Databricks-managed storage.

Solution: You create a Databricks access connector.

Does this meet the goal?

Options:

A- Yes
B- No

Answer:

B

Explanation:

CORRECT ANSWE R: B - No.

According to Microsoft Learn on Azure Databricks Access Connectors, a Databricks Access Connector is an Azure resource that allows Azure Databricks to authenticate to Azure Data Lake Storage Gen2 and other Azure services using a managed identity. An Access Connector is an authentication mechanism, not a data federation or catalog feature. Creating an Access Connector does not expose DB1's schemas and tables in Unity Catalog, does not enable querying of SQL Server data, and does not create any catalog representation of the external database. The Access Connector is a prerequisite for

setting up external locations or storage credentials in Unity Catalog, but it alone does not meet the federation requirements. To expose DB1 in Unity Catalog, a foreign catalog (Lakehouse Federation) is required, not an Access Connector.

Question 7

Question Type: MultipleChoice

Case Study: Mix Questions

Mix Questions

DP-750 Mix Questions IN THIS CASE STUDY

You have an Azure Databricks workspace named Workspace1 that contains a lakehouse and is enabled for Unity Catalog.

You have a connection to a Microsoft SQL Server database named DB1.

You need to expose the schemas and tables of DB1 to meet the following requirements:

* The schemas and tables can be queried in Databricks.

* The schemas and tables appear alongside other Unity Catalog objects.

* The data is NOT copied into Databricks-managed storage.

Solution: You create a foreign catalog in Catalog Explorer.

Does this meet the goal?

Options:

A- Yes
B- No

Answer:

A

Explanation:

CORRECT ANSWE R: A - Yes.

According to Microsoft Learn on Lakehouse Federation and Unity Catalog foreign catalogs, a foreign catalog is a Unity Catalog object that represents an external database (such as SQL Server) through a

registered connection. Creating a foreign catalog in Catalog Explorer using an existing connection to DB1 exposes all schemas and tables from DB1 as queryable objects within Unity Catalog. These objects appear alongside native Unity Catalog objects in Catalog Explorer. Crucially, Lakehouse Federation queries the external database in place --- the data is never copied into Databricks-managed storage. This satisfies all three requirements: schemas and tables can be queried in Databricks, they appear alongside other Unity Catalog objects, and the data is not copied. The foreign catalog is created under the registered connection to DB1.

Thank You for trying DP-750 PDF Demo

To try our DP-750 practice exam software visit link below

https://prepbolt.com/DP-750.html

Start Your DP-750 Preparation

Use Coupon “SAVE50” for extra 50% discount on the purchase of Practice Test Software. Test your DP-750 preparation with actual exam questions.